Does Malware Detection Improve With Diverse AntiVirus Products? An Empirical Study

Michel Cukier[1], Ilir Gashi[2], Betrand Sobesto[1], Vladimir Stankovic[2]

[1] University of Maryland, College Park, MD, USA
{mcukier, bsobesto}@umd.edu
[2]Centre for Software Reliability, City University London, London, UK
{i.gashi, v.stankovic}@csr.city.ac.uk

Abstract. We present results of an empirical study to evaluate the detection capability of diverse AntiVirus products (AVs). We used malware samples collected in a geographically distributed honeypot deployment in several different countries and organizations. The malware was collected in August 2012: the results are relevant to recent and current threats observed in the internet. We sent these malware to 42 AVs available from the VirusTotal service to evaluate the benefits in detection from using more than one AV. We then compare these findings with similar ones performed in the past to evaluate diversity with AVs. In general we found that the new findings are consistent with previous ones, despite some differences. This study provides additional evidence that detection capabilities are improved by diversity with AVs.

Keywords. Empirical study; Intrusion tolerance; Malware; Measurement techniques; Security; Security assessment tools.

1 Introduction

All systems need to be sufficiently reliable and secure in delivering the service that is required of them. Various ways in which this can be achieved in practice: from the use of various validation and verification techniques; to the use of software fault/intrusion tolerance techniques; or continuous maintenance and patching once the product is released. Fault tolerance techniques range from simple "wrappers" of the software components [1] to the use of diverse software products in a fault-tolerant system [2]. Implementing fault tolerance with diversity was historically considered prohibitively expensive, due to the need for multiple bespoke software versions. However, the multitude of available off-the-shelf software for various applications has made the use of software diversity an affordable option for fault tolerance against either malicious or accidental faults.

Authors in [3] detailed an implementation of an AntiVirus (AV) platform that makes use of diverse AVs for malware detection. A similar architecture that uses diverse AV email scanners has been commercially available for several years [4].

Thus, architectural solutions for employing diverse AV detection engines are already known and even commercially deployed. Results from empirical evaluation of the effectiveness of diversity for malware detection are, however, much more scarce.

The following claim is made on the VirusTotal site [5]: "*Currently, there is no solution that offers 100% effectiveness in detecting viruses, malware and malicious URLs*". Given these limitations of individual AVs, designers of security protection systems are interested in at least getting estimates of the possible gains in terms of added security that the use of diversity (e.g. diverse AVs) may bring for their systems.

Two of the authors of this paper have previously reported [6-8] results from a study on the detection capabilities of different AVs and potential improvements in detection that can be observed from using diverse AVs. In those studies we reported that some AVs achieved high detection rates, but none detected all the malware samples. We also found many cases of *regression* in the detection capability of the AVs: cases where an AV would regress from detecting the malware on a given date to not detecting the same malware at a later date(s). We reported significant improvements in the detection capability when using two or more diverse AVs. For example, even though no single AV detected all the malware in these studies, almost 25% of all the diverse 1-out-of-2 pairs of AVs successfully detected all the malware.

The results presented in [6-8] are intriguing. However, they concern a specific snapshot in the detection capabilities of AVs against malware threats prevalent in that time period: 1599 malware samples collected from a distributed honeypot deployment over a period of 178 days from February to August 2008. In the security field the threat landscape changes rapidly and it is not clear to what extent these findings can be generalized to currently spreading malware. It is also not clear whether the diversity benefits reported in [6-8] are specific to that specific collection environment and time period, or whether they are consistent with other environments and time periods.

Our work is motivated by the following claim from Fred Schneider in [9]: "*Experimentation is the way to gain confidence in the accuracy of our approximations and models. And just as experimentation in the natural sciences is supported by laboratories, experimentation for a science of cybersecurity will require test beds where controlled experiments can be run.*" In this paper we present results of an empirical study about possible benefits of diversity with currently spreading malware and compare our findings with those reported in [6-8]. The main aim of our study is to verify the extent to which the findings previously reported are relevant with more recent malware. Consistent with the statement in [9], through experimentation and empirical analysis our goal is to gain confidence in the accuracy of the claims and help security decision makers and researchers to make more informed, empirically-supported decisions about the design, assessment and deployment of security solutions.

The results provide an interesting analysis of the detection capability of the respective signature-based components, though more work is needed for assessing full detection capabilities of the AVs. Also, the purpose of our study is not to rank the individual AVs, but to analyze the effectiveness of using diverse AVs.

The rest of the paper is organized as follows: Section 2 summarizes related work; Section 3 describes the data collection infrastructure; Section 4 presents the results of our study; Section 5 compares the results reported in this paper with the ones in [6-8]; Section 6 discusses the implications of our results on the decision making about security protection systems, and presents conclusions and possible further work.

2 Related Work

Studies which perform analysis of the detection capabilities and rank various AVs are very common. One such study, which provides analysis of "at risk time" for single AVs, is given in [10]. Several sites[1] report rankings and comparisons of AVs, though readers should be careful about the definitions of "system under test" when comparing the results from different reports.

Empirical analyses of the benefits of diversity with diverse AVs are much less common. Apart from our previous work [6-8] (we refer to the findings reported there when we compare them with the new findings in Section 5) we know of only one other published study [3] that has looked at the problem. An initial implementation of the Cloud-AV architecture has been provided in [3], which utilizes multiple diverse AVs. The Cloud-AV uses the client-server paradigm. Each machine in a network runs a host service which monitors the host and forwards suspicious files to a centralized network service. This service uses a set of diverse AVs to examine the file, and based on the adopted security policy makes a decision regarding maliciousness of the file. This decision is then forwarded to the host. To improve performance, the host service adds the decision to its local repository. Hence, subsequent encounters of the same file by the host will be decided locally. The implementation from [3] handles executable files only. A study with a Cloud-AV deployment in a university network over a six month period is given in [3]. For the files observed in the study, the network overhead and the time needed for an AV to make a decision are relatively low. This is because the processes running on the local host, during the observation period, could make a decision in more than 99% of the cases. The authors acknowledge that the performance penalties could be much higher if file types other than just executables are examined, or if the number of new files observed on the host is high (since the host will need to forward the files for examination to the network service more often).

Apart from Cloud-AV, which is an academic prototype, commercial solutions that use diverse AV engines for file and e-mail scanning are also available[2].

3 Experimental Infrastructure

The malware have been collected on a distributed honeypot architecture using Dionaea - a low-interaction honeypot used to emulate common vulnerabilities, cap-

[1] av-comparatives.org , av-test.org/, virusbtn.com/index

[2] gfi.com/maildefense/ , pcworld.com/article/165600/G_data_internet_security.html

ture malicious payloads attempting to exploit the vulnerabilities and collect the binary files downloaded or uploaded. In summary, the main components of the experimental infrastructure and the process of data collection are as follows (full details are available in our technical report [11]):

- Dionaea has been deployed on 1136 public IP addresses distributed in six different locations in France, Germany, Morocco and the USA.
- The subnets do not contain the same number of IP addresses and the configuration differs between networks. Many IP addresses belong to University of Maryland (which is where two of the authors of this paper are based). Note that neither all networks apply the same security policies nor are protected in the same way.
- We deployed the default configuration of Dionaea which exposes common Internet services such as http, ftp, smtp, MS SQL, MySQL, as well as Microsoft Windows and VOIP protocols. These services and protocols emulate known vulnerabilities and can trap malware exploiting them. Due to the types of vulnerabilities and protocols emulated, Dionaea mainly collects Windows Portable Executable (PE) files.
- For this study, one Dionaea instance was deployed on each separate subnet, running on a different Linux virtual machine. To facilitate the analysis, all the malware collected by Dionaea and the information relative to their submission were merged and centralized on a single server.
- Every day at midnight a Perl script downloads from the virtual machines running Dionaea the binary files and the SQLite database containing the malware capture information. This script then submits the whole malware repository to VirusTotal [5], which is a web service providing online malware analysis based on several AVs. For each binary file successfully submitted, VirusTotal returns a scan key. The scan key is composed of the binary's SHA1 hash and the timestamp of the submission. To ensure a correct submission of each file and to later retrieve the analysis results the script keeps track of the scan keys.
- A second Perl script retrieves the reports for each malware using the scan keys generated by VirusTotal. VirusTotal returns an array containing: the number of AVs that have flagged the file as malicious, the total number of AVs used in the analysis, and the AVs names, versions and the signatures name.

4 Results

4.1 Basic Statistics from Our Results

Using the dataset introduced in Section 3 we explore the potential benefits in malware detection from employing diverse AVs. We start with some descriptive statistics of the obtained results.

Data collection lasted for 23 days: 8-30 August 2012. During this period we collected 922 malware. These malware were sent to VirusTotal where they were examined by up to 42 AVs. We sent the malware on the first day of observation and con-

tinued to send them throughout the collection period. However the total number of data points we have is not simply 23 * 922 * 42. It is smaller because:

- not all malware were observed in the first day of collection – we continued to observe new malware throughout the collection period and we cannot send a newly collected malware to older versions of AVs running on VirusTotal;
- VirusTotal may not always return results for all AVs – we are not sure why this is. VirusTotal is a black-box service and its internal configuration is not provided. We presume that each AV is given a certain amount of time to respond; if it doesn't, VirusTotal will not return a result for that AV. Additionally a particular AV may not be available at the time we submit the malware for inspection.

A unique "demand" for the purpose of our analysis is a {$Malware_j$, $Date_k$} pair which associates a given malware *j* to a given date *k* in which it was sent to VirusTotal. We treat each of the malware sent on a different date as a unique demand. If all 922 malware were sent to VirusTotal on each of the 23 days of data collection, then we would have 922 * 23 = 21,206 demands. But as explained before, due to missing data, the number of demands sent to any of the AVs is smaller than 21,206.

If we now associate a given AV *i*'s response to a given malware *j* on a given date *k* then we can consider each of our data points in the experiment to be a unique triplet {AV_i, $Malware_j$, $Date_k$}. For each triplet we have defined a binary score: 0 in case of successful detection, 1 in case of failure. Table 1 shows the aggregated counts of the 0s and 1s for the whole period of our data collection. We have considered as success the generation of an alert by an AV regardless of the nature of the alert itself.

Table 1 - Counts of detetcions and failures for triplets {AV_i, $Malware_j$, $Date_k$ }

Value	*Count*
0 – detection / no failure	*766,853*
1 – no detection / failure	*65,410*

4.2 Single AV Results

Table 2 contains the failure rates of all the 42 AVs. The ordering is by the failure rate (second column) with the AV with the smallest failure rate appearing first.

The third column in Table 2 counts the number of "releases" of a given AV recorded by VirusTotal. We presume these are the versions of either the rule set or the release version of the detection engine itself. It seems that different products have different conventions for this. Amongst the three equally best AVs in our study, Ikarus reports only one version whereas AntiVir and ESET-NOD32 have 36 and 27 sub-release versions respectively.

The fourth and fifth columns of Table 2 report on an interesting phenomenon first reported in our previous work [8] – some AVs *regress* on their detection capability. That is, they detected the malware at first, and then failed to detect the malware at a

Table 2. Failure Rates for Each AV

AV Name	Failure rate	Number of "releases" of the AV in VirusTotal	Count of Malware on which AV regressed	Count of Regression Instances
AntiVir	0.000049	1	0	0
ESET-NOD32	0.000049	36	0	0
Ikarus	0.000049	27	0	0
Kaspersky	0.000050	1	1	1
Sophos	0.000050	2	0	0
VIPRE	0.000098	29	0	0
McAfee	0.000099	1	0	0
Norman	0.000099	1	0	0
Emsisoft	0.000208	1	3	3
Symantec	0.001112	2	0	0
F-Secure	0.001204	1	0	0
Avast	0.001211	1	0	0
BitDefender	0.001232	1	0	0
PCTools	0.001244	1	0	0
Jiangmin	0.001286	1	0	0
AVG	0.001342	1	0	0
GData	0.001627	1	8	8
TrendMicroHouseC.	0.001847	2	13	13
K7AntiVirus	0.001881	19	0	0
McAfee-GW-Ed.	0.002232	1	43	43
VirusBuster	0.002256	27	0	0
nProtect	0.002503	26	0	0
Microsoft	0.002515	3	0	0
TheHacker	0.002522	1	1	1
TrendMicro	0.002530	2	26	26
DrWeb	0.003517	2	0	0
ViRobot	0.003518	1	0	0
Panda	0.003530	1	4	17
TotalDefense	0.003708	23	0	0
VBA32	0.006567	2	55	55
Comodo	0.009233	32	139	151
CAT-QuickHeal	0.010306	1	1	1
AhnLab-V3	0.010950	25	105	120
F-Prot	0.011789	1	1	1
Commtouch	0.012832	1	1	1
eSafe	0.165981	1	8	9
Rising	0.226335	18	10	10
SUPERAntiSpyware	0.356355	2	0	0
ClamAV	0.377830	1	0	0
Antiy-AVL	0.412142	1	1	1
Fortinet	0.670168	1	5	5
ByteHero	0.963825	1	33	59

later date(s), probably due to some updates in the respective AV's rule definitions. The fourth column contains the number of malware on which a given AV regressed, and the fifth column contains the number of instances of these regressions (since an AV may have regressed more than once on a given malware: e.g. alternated between detection and non-detection of a malware several times). We note that even a few AVs, which are in the top ten in terms of the overall detection rates, did have cases of regressions. Such a phenomenon can be due to various reasons. For instance, the vendor might have deleted the corresponding detection signature as a consequence of the identification of false positives associated to it, or they might be attempting to consol-

idate and streamline the signature based detection rules (i.e. define a smaller number of more generic rules) to achieve faster detection.

Even though some of the AVs have very good detection rates, none of them have detected all the malware in our study. We can also see that some AVs have really low detection rates, with the bottom 7 AVs failing to detect more than 10% of all the demands sent to them. We are not certain why this is the case. It could be because the AV vendors are failing to keep their AVs in VirusTotal up to date with their latest signatures even if their products in commercial installations are up to date (though some of these vendors do seem to be updating their products with new release numbers, as evidenced from the values in the third column). Alternatively, it could be because these AVs genuinely have low detection rates for this dataset.

In our previous work [6] we have also observed a similar phenomenon: six of the AVs had failure rates lower than 10%. We cannot report if any of these 6 AVs are in the subset of the 7 AVs above, since in [6] the AV names were anonymised. Before we proceeded with the diversity analysis in [6] we discarded from the diversity analysis the AVs that had failure rates lower than 10%. We justified this by stating "*It is inconceivable that any system administrator would choose AVs that have such a bad detection rate to be used in their system. Hence we decided to remove the 6 worst performing AVs from further analysis. The decision was also influenced by our goal to make any results of the benefits of diversity appear fairer. A criticism that can be made if these 6 worst performing AVs are included in the analysis is that improvements in detection capability through the use of diversity will of course be higher if you have such poorly performing individual AVs in the mix. By removing them we make our estimates of the benefits of detection via diversity more conservative*".

To keep a consistent comparison of the results in this paper with those reported in [6-8] we discarded the AVs with failure rates greater than 10% from the diversity analysis. The rest of the analysis is based on the best 35 AVs from Table 2.

4.3 Diverse AV Results

To evaluate the possible benefits in detection capabilities that using diverse AVs may bring, we looked at two different types of diversity configurations/setups:

- *1-out-of-N* (or *1ooN*), where N is the total number of AVs in a given configuration – a malware is deemed to have been detected on a given date as long as at least one of the AVs detected it.
- *r-out-of-N* (or *rooN*) where N is the total number of AVs in a given configuration and *r* is the minimum number of AVs that must detect a malware on a given date for it to be deemed as detected. We looked at two particular voting options:
  - *majority voting*: where N is an odd number and *r* is (N+1)/2 - this allows a tie-breaker via majority voting;
  - *trigger level:* where a fixed number *r* of AVs have to detect a malware (i.e. "be triggered") before the malw*are is consid*ered to be detected – we will present results when the trigger level *r* equals 2 or 3.

Of course many other voting configurations are possible, but we chose these two above as they best represent the contrasting tradeoffs between detection rates and false alarm rates that decision makers should take into consideration when deciding on a diverse system configuration: a 1ooN configuration could be used to maximize the detection rates of genuine malware; a majority voting one could be used to curtail the false positive rate; the trigger level configurations allow a trade-off between these two. Note that since we are dealing with confirmed malware samples we cannot really present any data about false positives. However, the rooN results allow us to get initial estimates of how is the rate of correct detections of confirmed malware affected if one uses majority voting or trigger-level setup to reduce the false positive rate.

Due to space limitations we only present a snapshot of the results we obtained. We concentrated on showing a graphical representation of the results. The details and the tables from which these graphs are generated as well as other detailed results are available in [11]. The cumulative distribution functions (*cdf*) of the failure rate achieved for 1ooN, 2ooN, 3ooN and majority voting setups are shown in Figure 1.

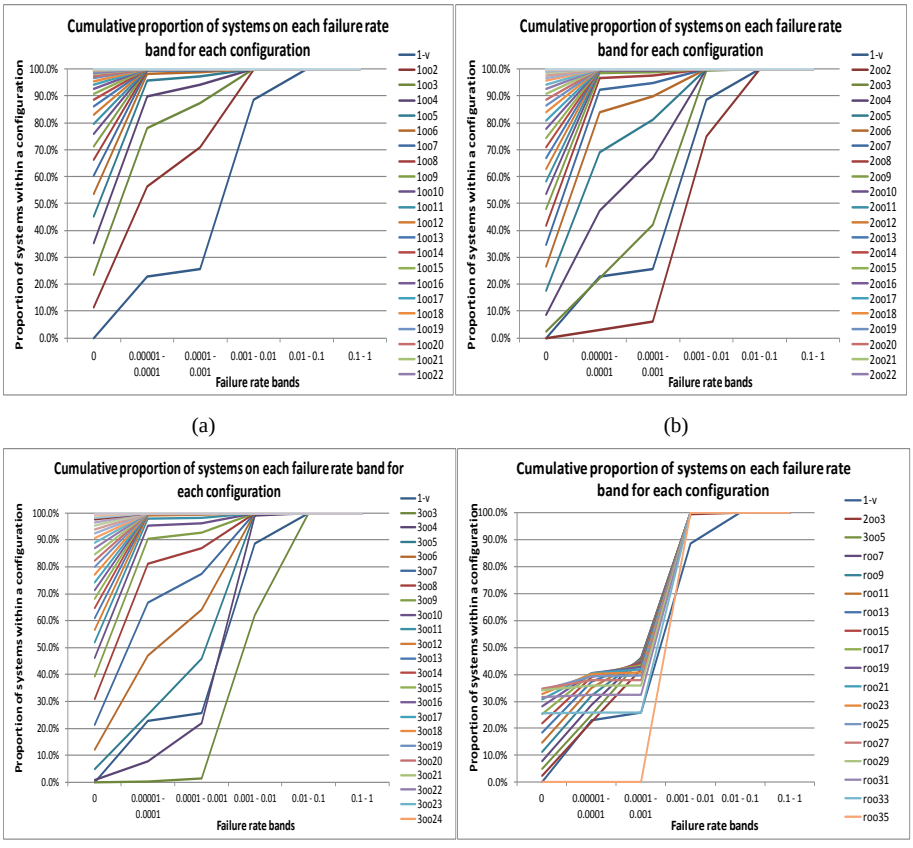

Fig. 1. Cumulative distribution of failure rate for a)*1ooN*, b)*2ooN*, c)*3ooN*, and d) *rooN* setups.

For part (a), (b) and (c) of Figure 1 the legend shows configurations up to N = 22 or 24, but in fact all 1ooN diverse configurations are contained in the graph. For 1ooN, 2ooN and 3ooN configurations the figures visualize the trend of improving failure rates as we add more AV products in diverse configurations. The shape of the distributions for the majority voting setup (part (d) of Figure 1) is different and we see a decline in the proportion of perfect systems after N=29.

Figure 2 presents the difference in the proportion of the non-perfect detection combinations in these three different setups. We can see that to get 90% of all 1ooN systems to have a perfect detection of all malware with our dataset we need N to be between 14 and 15 (we can observe this in Figure 2 by following where the x-axis value 14 and 15 meets the y-axis value 1.E-01 for the 1ooN line). To get 90% of 2ooN systems to detect all malware we need N~20 and 21, whereas N for 3ooN is between 23 and 24. This is another way in which an administrator could measure the added cost of seeking confirmation from 2 or 3 AVs before raising alarms.

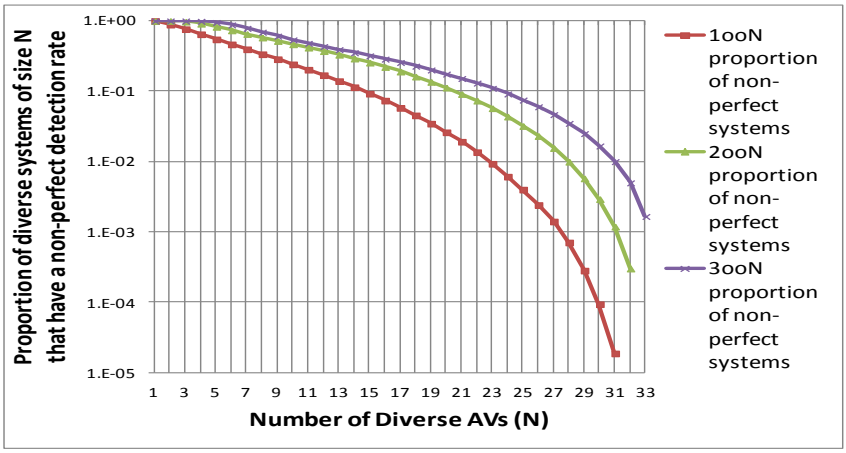

Fig. 2. Proportion of diverse systems of size N that have a non-perfect detection rate.

5 Discussion and Comparison of Results With Other Studies

We now discuss and compare the results of the study presented in this paper with those from our previous study [6-8]. In summary:

- In [6-8] despite the generally high detection rates of the AVs, none of them achieved 100% detection rate.
  Our observation: we also observe this with the new dataset. Eight of the AVs in our study had failure rates smaller than 1.E-04 but none of them detected all the instances of malware (on all days) in our study either.
- In [6-8] the detection failures were both due to an incomplete signature databases at the time in which the samples were first submitted for inspection, but also due to

regressions in the ability to repeatedly detect malware as a consequence, possibly, of the deletion of some signatures.
Our observation: we also observe this with the new dataset. Regressions in the detection capabilities were observed even with AVs who are ranked in the top 10 in our dataset in terms of their detection capability.

- In [6-8] considerable improvements in detection rates were observed from employing diverse AVs: almost 25% of all the diverse pairs, and over 50% of all triplets in respective 1-out-of-N configurations successfully detected all the malware.
  Our observation: we observe improvements in 1-out-of-N configurations though not as good as those in [6-8]: under 12% of all diverse pairs, and under 24% of all diverse triplets detected all the malware in 1-out-of-N configurations. To get over 50% of all combinations to detect all the malware we need 6 AVs rather than 3.
- In [6-8], no malware caused more than 14 AVs to fail on any given date. Hence perfect detection rates, with their dataset is achieved by using 15 AVs in a 1-out-of-N configuration.
  Our observation: We had malware that failed to be detected by up to 31 AVs in the new dataset. So to get perfect detection rates with our dataset for all possible instances of a diverse configuration, we need 32 AVs.
- In [7] the detection rates were lower for "trigger level detection" (2-out-of-N and 3-out-of-N) and majority voting (r-out-of-N) setups compared with 1-out-of-N but on average are better than using a single AV.
  Our observation: We confirm that these findings are consistent with what we found previously. Additionally we found that the proportion of perfect majority voting systems is considerably higher with the new dataset compared with those reported in [7]: in the new analysis more than 34% of possible combinations of 14oo27 and 15oo29 majority voting configurations achieved perfect detection. In [7] the proportion of perfect detection majority voting systems never reached more than 6% for any of the combinations in that study. This may have implications on the choice of diverse setups that administrators may use especially if false positives become an issue. If the detection capability against genuine malware is not (significantly) hampered from a majority decision by diverse AVs, then majority voting setups could become more attractive to administrators than 1ooN configurations.
- In [6] significant potential gains were observed in reducing the "at risk time" of a system from employing diverse AVs: even in cases where AVs failed to detect a malware, there is diversity in the time it takes different vendors to define a signature to detect a malware.
  Our observation: This is also supported by the results in our study (though due to space limitations we could not elaborate on this here; see [11] for details).
- In [6] an empirically derived exponential power law model proved to be a good fit to the proportion of systems in each simple detection (1-out-of-N) and trigger level detection (2-out-of-N and 3-out-of-N) diverse setup that had a zero failure rate.
  Our observation: we have not explored the modeling aspects in this paper. There do appear to be some differences between the shapes of the empirical distributions of the proportion of diverse systems of size N that have a non-perfect detection rate (see Figure 2) from what is presented in [6]. The empirical distributions in Figure 2

do look like they follow an exponential power law model (possibly of a different form from that observed in [6]), so further work is needed to check whether the model outlined in [6] can be generalized for this dataset. A generalized model would allow a cost-effective prediction of the probability of perfect detection for systems that use a large number of AVs based on measurements made with systems that are composed of fewer (say 2 or 3) AVs.

6 Discussion and Conclusions

We reported analysis of results from an empirical study on the possible benefits of diversity with currently spreading malware and compared and contrasted the findings with those we reported previously using a different dataset [6-8]. We verified the extent to which the findings previously reported in [6-8] are relevant with more recent malware. The new results were in general consistent with those reported in [6-8] though there were differences. The consistent results were:

- None of the single AVs achieved perfect detection rates
- A number of the AVs regressed in their detection behavior (i.e. failed to detect malware which they had successfully detected in the past)
- Considerable improvements in detection capability when using diversity
- As one would expect there is an ordering in the effectiveness of detection capability with 1-out-of-N, 2-out-of-N, 3-out-of-N and majority voting diverse setups, with 1-out-of-N having the best detection rates
- Using diverse AVs helps with reducing the "at risk time" of a system

The main differences in the results were:

- Despite significant improvements from diversity with 1-out-of-N, 2-out-of-N and 3-out-of-N, they are lower than those reported in [6-8]. For example with 1ooN we observed that 12% of all diverse pairs and 24% of all diverse triplets detected all the malware (compared with approximately 25% and 50% respectively in [6-8]).
- On the other hand, we observe a much higher proportion of perfect detection majority voting systems in all setups compared with the results observed in [6-8]
- The shape of the empirical distribution of non-perfect detection systems as more diverse AVs are added seems different from that observed in [6] and should be investigated further.

The aim of this work is to provide more evidence on the possible benefits of using diverse AVs to help with malware detection and therefore help security decision makers, and other researchers in the field to make more informed, empirically-supported decisions on the design, assessment and deployment of security protection systems.

The limitations to this work which prevent us from making more general conclusions and hence require further work are as follows. First, we have no data on false positives. Hence studying the detection capabilities with datasets that allow measurements of false positives in addition to false negative rates will allow us a better analy-

sis of the tradeoffs between the various 1-out-of-N, trigger-level and majority voting detection setups. Second, we have only tested the detection capability when subjecting the AVs to Windows portable executable files. Further studies are needed to check the detection capability for other types of files e.g. document files, media files etc. Third, the AV products contain more components than just the signature-based detection engine which is what the VirusTotal service provides. Further studies are needed to include the detection capabilities of these products in full. Fourth, due to lack of space in this paper, we have not explored the modeling aspects and modeling for prediction. Our next step is to do further work in checking whether a form of the power law distribution observed in [6] also applies with this dataset.

Current work in progress includes: the patterns with which AVs label the malware when they detect them. We have started doing this analysis to study the patterns of label changes, whether they differ for different products, and whether we can show a causal link between frequent malware label changes and AVs regressing on their detection capability. In addition, each of the malware in our sample has a different MD5 hashvalue, but it is not clear how similar/different they are. So we are investigating whether polymorphic malware is identified with the same labels by the AVs and whether this can aid with diagnosis and recovery from malware infections.

References

1. van der Meulen, M.J.P., S. Riddle, L. Strigini and N. Jefferson. "Protective Wrapping of Off-the-Shelf Components". in the 4th Int. Conf. on COTS-Based Software Systems (ICCBSS). Bilbao, Spain, p. 168-177, 2005.
2. Strigini, L., "Fault Tolerance against Design Faults", in Dependable Computing Systems: Paradigms, Performance Issues, and Applications, H. Diab and A. Zomaya, Editors, J. Wiley & Sons. p. 213-241, 2005.
3. Oberheide, J., E. Cooke and F. Jahanian. "Cloudav: N-Version Antivirus in the Network Cloud". in the 17th USENIX Security Symposium, p. 91–106, 2008.
4. GFi. Gfimaildefence Suite, last checked 2012, http://www.gfi.com/maildefense/.
5. VirusTotal. Virustotal - a Service for Analysing Suspicious Files, last checked 2013, http://www.virustotal.com/sobre.html.
6. Bishop, P., R. Bloomfield, I. Gashi and V. Stankovic. "Diversity for Security: A Study with Off-the-Shelf Antivirus Engines". in the 22nd IEEE International Symposium on Software Reliability Engineering (ISSRE), p. 11-19, 2011.
7. Bishop, P.G., R.E. Bloomfield, I. Gashi and V. Stankovic, "Diverse Protection Systems for Improving Security: A Study with Antivirus Engines". 2012, City University London: London, UK.
8. Gashi, I., C. Leita, O. Thonnard and V. Stankovic. "An Experimental Study of Diversity with Off-the-Shelf Antivirus Engines". in the 8th IEEE Int. Symp. on Network Computing and Applications (NCA 2009), p. 4-11, 2009.
9. Schneider, F., "Blueprint for a Science of Cybersecurity". The Next Wave, 19(2): p. 47-57. 2012.
10. Sukwong, O., H.S. Kim and J.C. Hoe, "Commercial Antivirus Software Effectiveness: An Empirical Study". IEEE Computer, 44(3): p. 63-70. 2011.
11. Cukier, M., I. Gashi, B. Sobesto and V. Stankovic, "Technical report: Does Malware Detection Improve with Diverse Antivirus Products? An Empirical Study", 2013, http://www.csr.city.ac.uk/people/ilir.gashi/SAFECOMP2013/